REMARKS

In the Office Action mailed March 20, 2006,[1] the Examiner provisionally rejected claims 1-27, on the ground of nonstatutory obviousness-type double patenting, as being unpatentable over claims 1-25 of copending Application No. 09/760,031; withdrew the rejection of claim 28 under 35 U.S.C. § 101 (because Applicants cancelled claim 28); and rejected claims 1-27 under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 5,999,948 to Nelson et al. ("Nelson"). In response to the Office Action, Applicants respectfully request reconsideration. Claims 11-27 and 29-30 are pending in the application of which claims 1, 13, 25, and 26 are independent. Claims 1, 5, 13, 23, 25, and 26 are amended in the forgoing amendments.

Applicants have amended claims 1, 5, 13, 25, and 26 to clarify that the claimed methods use object oriented programming and that allocation of memory space occurs after the creation of an instance of a class. Applicants have also added claims 29 and 30 to further describe Applicants' invention.

Response to Arguments

First, Applicants respectfully disagree with Examiner's characterization of Applicants' invention on page 3 to page 4 of the Office Action. Applicants agree that the claims relate to "an object oriented language, using a class template type of structure which includes optional elements (option data structure) stored in a linked list or another type of data structure." (Office Action at 3). However, Applicants disagree with the statements that follow in which the Examiner states that memory is dynamically

[1] The Office Action may contain statements characterizing the related art, case law, and claims. Regardless of whether any such statements are specifically identified herein, Applicants decline to automatically subscribe to any statements in the Office Action.

allocated *at compilation* as needed. (Office Action at 3, emphasis added). Instances of a class are not created at compilation but at runtime. In Applicants' invention, memory is dynamically allocated for full option values after the creation of an instance of the class. For example, in the preferred embodiment: "The size of each instance is only proportional to the number of option values that have been set on that instance" (Detailed Description, paragraph 0068). Moreover, in the embodiment, allocation of memory for the full option values occurs when the option value is set which cannot occur until after an instance is created. (*Id.*) Since the instance is not created until after compilation, and a set operation on an instance cannot occur until after the creation of an instance, the dynamic allocation of memory for option values is not performed at compilation as the Examiner has alleged.

Applicants also disagree that "dynamic allocation of memory in an object oriented environment, using type checking is well known in the art." Applicants maintain that type checking of a reference to an option value within a data structure during compilation, as defined in the specification, is not well known in the art. Applicant challenges this reliance on common knowledge and requests that the Examiner produce authority for this statement. (See M.P.E.P. § 2144.03(C)).

Provisional Nonstatutory Obviousness-type Double Patenting

In the Office Action mailed March 20, 2006, the Examiner provisionally rejected claims 1-27 under the judicially created doctrine of obviousness-type double patenting over claims 1-25 of copending Application No. 09/760,031. Applicants request that the Examiner holds the double patenting rejection in abeyance until claims 1-27 and 29-30 are otherwise allowable.

Rejection of Claims 1-27 Under 35 U.S.C. § 102(e)

Applicants respectfully traverse the rejection of claims 1-27 under 35 U.S.C. § 102(e) as being anticipated by Nelson. To properly establish that a prior art reference anticipates a claimed invention under 35 U.S.C. § 102, each and every element of the claims in issue must be found, either expressly described or under principles of inherency, in the single prior art reference. Applicants submit that independent claim 1 is not anticipated by Nelson because the reference fails to teach each and every claim element of the claim.

Nelson discloses a declarative language for displaying a form (defining what a form will look like) and not an imperative, object oriented language where programs for performing algorithms and processing data are written using objects that act on one another. Specifically, Nelson at least fails to disclose "compiling an operation on an option value in an instance of the class using the type description in the option data structure." Compilation is the process by which a compiler translates program code into machine language, or object code. Ivor Horton, Beginning Visual C++ 6 p.10 (Wrox Press Ltd. 1998). During this process, errors are detected due to invalid or unrecognized program code or structural errors. (*Id.*)

As a disclosure of the compilation stage in the data processing method of claim 1, the Examiner cites Nelson, Col. 12, lines 61-67 which states, "During the exchange, the DataItem for each widget is looked up in the Datacollection and the appropriate read/write function is executed. A read gets data from the DataItem and places it in the widget and a write sets data from the widget into the DataItem." (Office Action at 6). This citation does not disclose or suggest the conversion of program code

to object code consistent with the definition of compilation in an object oriented programming language.

In addition, Nelson does not disclose or suggest "in *instances* of the class, references to option values without allocation of memory space for the full option values when the instance is created". (emphasis added). The Examiner equates "class" as used in the claims with "Form specification class" in Nelson. (Office Action at 5). Nowhere in Nelson does it demonstrate that an *instance* of the form specification class has "references to option values without allocation of memory space for the full option values when the instance is created."

For the reasons stated above with respect to claim 1, claims 1, 13, 25, and 26 which recite similar limitations relating to failing to allocate memory when an instance is created, and compilation, and claims 2-12, and 14-24 which depend therefrom, are allowable.

In view of the foregoing amendments and remarks, Applicants respectfully request reconsideration and reexamination of this application and the timely allowance of the pending claims.

Please grant any extensions of time required to enter this response and charge any additional required fees to our deposit account 06-0916.

Respectfully submitted,

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER, L.L.P.

Dated: September 20, 2006

By: _____
Christopher S. Schultz
Reg. No. 37,929